LAPORAN PRAKTIKUM PEKAN 4

ALGORITMA DAN PEMROGRAMAN

CONDITIONAL STATEMENT

Disusun oleh:

Thaariq Salam

2511532022

Dosen Pengampu: Dr. Wahyudi S.T.M.T

Asisten Praktikum: Rahmad Dwirizki Olders

DEPARTEMEN INFORMATIKA

FAKULTAS TEKNOLOGI INFORMASI

UNIVERSITAS ANDALAS

TAHUN 2025

KATA PENGANTAR

Puji syukur kami panjatkan kehadirat Allah SWT atas rahmat dan karunia-Nya, sehingga Laporan Praktikum mata kuliah Pemrograman Algoritma dan Pemrograman dengan topik "Struktur Kontrol: Pernyataan Kondisi (*Conditional Statement*)" ini dapat diselesaikan secara komprehensif. Dokumen ini disusun sebagai pemenuhan kewajiban akademis pada Praktikum Pekan 4.

Praktikum ini berfokus pada d*if*erensiasi dan implementasi berbagai konstruksi kondisi, mulai dari pernyataan *if* hingga *switch-case*. Penguasaan materi ini merupakan langkah fundamental untuk merancang alur program yang adaptif dan responsif terhadap variasi data masukan, yang merupakan inti dari logika pemrograman.

Penyusun menyadari bahwa keberhasilan penyusunan laporan ini tidak lepas dari bimbingan dan dukungan berbagai pihak. Oleh karena itu, penyusun menyampaikan terima kasih yang sebesar-besarnya kepada:

1. Bapak Dr. Wahyudi S.T.M.T selaku Dosen Pengampu mata kuliah Algoritma dan Pemrograman.
2. Rahmad Dwirizki Olders selaku Asisten Praktikum yang telah memberikan arahan teknis selama pelaksanaan praktikum.

Penyusun berharap laporan ini tidak hanya memenuhi aspek administratif, tetapi juga dapat menjadi dokumentasi yang bermanfaat dan media *review* bagi penyusun maupun pembaca lainnya. Kritik dan saran konstruktif sangat diharapkan demi perbaikan di masa mendatang.

Padang, 27 September 2025

Penulis

DAFTAR ISI

BAB I

PENDAHULUAN

1.1 Latar Belakang

Pada tahap awal pembelajaran pemrograman, instruksi kode cenderung dieksekusi secara linear atau berurutan dari atas ke bawah. Namun, dalam aplikasi nyata, sebuah program harus memiliki kemampuan untuk mengambil keputusan dan menjalankan alur tindakan yang berbeda berdasarkan data masukan yang bervariasi.

Praktikum ini didasari oleh kebutuhan untuk menguasai Struktur Kontrol Kondisi (*Conditional Statement*) sebagai pilar utama dalam logika pemrograman. Konstruksi kondisi, meliputi *if*, *if-else*, dan *switch-case*, berfungsi sebagai mekanisme yang memungkinkan program bertransformasi dari eksekusi statis menjadi dinamis dan fleksibel.

Melalui praktikum ini, mahasiswa dilatih untuk merancang logika perbandingan yang melibatkan evaluasi *Boolean* (*true* atau *false*), sehingga program mampu menyeleksi dan menentukan langkah yang paling tepat. Penguasaan atas materi ini adalah prasyarat fundamental untuk membangun algoritma yang tidak hanya berjalan lurus, tetapi juga adaptif dan responsif terhadap lingkungan pengguna.

1.2 Tujuan

Pelaksanaan praktikum pekan 4 ini punya target utama agar mahasiswa bisa:

1. Menguasai implementasi pernyataan *if* dan *if-else* guna memfasilitasi keputusan biner (dua arah) dalam alur program.

2. Mampu menyusun pernyataan berantai *if-else if-else* untuk melakukan pengujian multikriteria secara sekuensial dan efisien pada kondisi yang bersifat hierarkis
3. Memanfaatkan pernyataan *switch-case* sebagai alternatif yang lebih rapi untuk perbandingan nilai, serta memahami pentingnya penggunaan kata kunci break dan default.
4. Meningkatkan kemampuan analisis logika untuk merancang solusi pemrograman yang adaptif terhadap variasi input data.

1.3 Manfaat

Praktikum mengenai pernyataan kondisi ini memberikan kontribusi yang signifikan dalam pengembangan kemampuan teknis dan logis mahasiswa. Manfaat utamanya adalah peningkatan dalam kemampuan logika pengambilan keputusan, di mana mahasiswa dilatih untuk merumuskan solusi pemrograman yang adaptif terhadap variasi data masukan.

Penguasaan atas struktur kontrol, seperti *if-else if* dan *switch-case*, merupakan fondasi yang wajib dikuasai untuk menciptakan kode yang efisien, terstruktur, dan mudah dipelihara. Selain itu, kompetensi ini adalah bekal penting untuk membangun program yang tidak hanya mengeksekusi perintah secara berurutan, tetapi juga mampu berinteraksi secara cerdas dan respons*if*, menjadikan mahasiswa siap untuk merancang aplikasi yang kompleks dan dinamis.

BAB II

PEMBAHASAN

2.1 Kondisi bersyarat tunggal

2.1.1 Kode program

Praktikum ini berfungsi sebagai pengantar untuk memahami bagaimana program mulai mengambil keputusan dengan hanya mengeksekusi satu blok kode jika prasyarat terpenuhi.

```
package pekan4;
import java.util.Scanner;
public class lat If1 {
    public static void main(String[]args) {
            double IPK;
            Scanner input=new Scanner(System.in);
            System.out.print("Input IPK Anda = ");
            IPK=input.nextDouble();
            input.close();
            if (IPK >2.75) {
                    System.out.print("Anda lulus Sangat Memuaskan dengan
IPK" + IPK);
            }
}
}
```

2.1.2 Langkah kerja

1) Buat berkas Java bernama lat*If*1.java dan siapkan *class Scanner* untuk menerima input.
2) Minta pengguna memasukkan nilai IPK (double).
3) Definisikan pernyataan *if* tunggal untuk mengecek kondisi IPK > 2.75.

4) Sisipkan perintah *System.out.print* di dalam blok *if* sebagai pesan output jika syarat lulus terpenuhi.
5) Uji coba program: Amati bahwa ketika input IPK di bawah 2.75, program tidak menghasilkan output apa pun, namun ketika IPK di atas 2.75, output akan tercetak.

2.1.3 Analisis hasil

Kode ini menunjukkan penerapan pernyataan *if* tunggal yang berfungsi seperti gerbang tol satu arah. Program hanya akan menjalankan instruksi di dalamnya jika kondisi di dalam kurung bernilai *true* [1]. Ketika IPK yang dimasukkan gagal memenuhi syarat *(false),* program secara sederhana akan melewati seluruh blok *if* tanpa memberikan respons. Struktur ini merupakan konsep dasar dalam *flow control*, tetapi memiliki keterbatasan karena tidak menyediakan *fallback* atau tanggapan untuk kasus kegagalan

2.2 Kondisi Dua Arah (*if – else*)

2.2.1 Kode program

Praktikum ini mengatasi keterbatasan *if* tunggal dengan menyediakan jalur alternatif, menjamin program selalu memberikan respons.

```
package pekan4;
import java.util.Scanner;
public class ifelse1 {
    public static void main(String[]args) {
        double IPK;
        Scanner input=new Scanner(System.in);
        System.out.print("Input IPK Anda = ");
        IPK=input.nextDouble();
        input.close();
        if (IPK >2.75) {
```

```
        System.out.println("Anda lulus Sangat Memuaskan dengan IPK" + IPK);
        }
        else {
        System.out.println("Anda tidak lulus");
        }
}
}
```

2.2.2 Langkah kerja

1. Gunakan kode dari Praktikum 1 dan tambahkan pernyataan ***else***.
2. Di dalam blok ***else***, masukkan perintah output untuk kasus kegagalan ("Anda tidak lulus").
3. Uji coba program dengan input IPK yang tinggi (misalnya 3.5) dan rendah (misalnya 2.0).

2.2.3 Analisis hasil

Penggunaan struktur *if-else* menyempurnakan program menjadi sistem pengambilan keputusan biner yang lengkap [1]. Struktur ini menjamin bahwa satu dan hanya satu blok kode yang akan dijalankan. Jika kondisi di *if* tidak terpenuhi *(false),* kontrol program secara otomatis dialihkan ke blok *else*. Konsep ini sangat vital karena menghilangkan kemungkinan program "diam" dan memastikan selalu ada umpan balik yang jelas kepada pengguna, apapun input yang dimasukkan.

2.3 Kondisi Majemuk dan Operator Logika (*if* Majemuk)

2.3.1 Kode program

Praktikum ini fokus pada pengujian dua persyaratan atau lebih dalam satu evaluasi logis menggunakan operator &&.

```
package pekan4;

import java.util.Scanner;
```

```
public class multiIf {
    public static void main(String[]args) {
    int umur;
    char sim;
    Scanner a= new Scanner(System.in);
    System.out.print("Input umur anda :");
    umur= a.nextInt();
    System.out.print("Apakah anda sudah puhnya SIM? ");
    sim=a.next().charAt(0);
    a.close();
        if ((umur >= 17)&& (sim=='y')) {
            System.out.print("Anda sudah dewasa dan boleh bawa motor");
        }
        if ((umur >= 17)&& (sim!='y')) {
            System.out.print("Anda sudah dewasa tetapi tidak boleh bawa motor");
        }
        if ((umur <= 17)&& (sim!='y')) {
            System.out.print("Anda belum cukup umur bawa motor");
        }
        if ((umur <= 17)&& (sim=='y')) {
            System.out.print("Anda belum cukup umur punya SIM");
        }
```

```
        }
}
```

2.3.2 Langkah kerja

1) Buat berkas multi*If*.java dan deklarasikan variabel umur dan sim.
2) Tuliskan empat pernyataan *if* independen yang masing-masing menguji kombinasi dari umur dan sim.
3) Gunakan Operator Logika *AND (&&)* di setiap kondisi, karena semua syarat harus terpenuhi secara bersamaan.
4) Uji coba program dengan memasukkan nilai yang memicu setiap skenario (misalnya: dewasa dan punya SIM; belum dewasa dan tidak punya SIM).

2.3.3 Analisi hasil

Inti dari praktikum ini adalah penggunaan Operator Logika *AND (&&)* yang mengharuskan kedua sisi ekspresi bernilai *true* agar blok kode dieksekusi [1]. Meskipun semua skenario logis tercakup, penggunaan empat *if* yang terpisah untuk kondisi yang saling eksklus*if* adalah pendekatan yang tidak ideal. Sistem dipaksa untuk mengevaluasi semua empat kondisi secara berurutan, bahkan setelah kondisi yang benar sudah ditemukan. Ini menimbulkan *overhead* komputasi yang dapat dihindari, dan dalam praktiknya, skenario seperti ini lebih baik diatasi dengan struktur *if-else if-else* yang lebih ringkas dan efisien.

2.4 Kondisi Berjenjang/Bersarang/Nested

2.4.1 Kode program

```
package pekan4;

import java.util.Scanner;

public class Nilai {
```

```
    public static void main(String[]args) {
        int nilai;
        Scanner input= new Scanner(System.in);
        System.out.print("Input nilai angka= ");
        nilai= input.nextInt();
        input.close();

        if (nilai >=81) {
            System.out.print("A");
        } else if (nilai >=70) {
            System.out.print("B");
        } else if (nilai >=60) {
            System.out.print("C");
        } else if (nilai >=50) {
            System.out.print("D");
        } else {
            System.out.print("E");
        }
    }
}
```

2.4.2 Langkah kerja

1) Buat berkas Nilai.java dan terima input nilai angka.
2) Tuliskan pernyataan *if* awal untuk batas nilai tertinggi (>=81).

3) Lanjutkan dengan beberapa pernyataan *else if* yang disusun secara berurutan menurun (dari 70, 60, dan seterusnya).
4) Akhiri seluruh struktur dengan satu pernyataan *else* untuk menangani semua nilai di bawah batas terendah (nilai 'E').
5) Uji coba: Masukkan nilai seperti 75; amati bahwa program mencetak 'B' dan tidak perlu memeriksa kondisi di bawahnya.

2.4.3 Analisis hasil

Struktur *if-else if-else* adalah pilihan terbaik untuk mengklas*if*ikasikan data berdasarkan jenjang atau rentang nilai, seperti pada sistem konversi nilai ini [1]. Keunggulan utamanya terletak pada efisiensinya. Begitu satu kondisi terpenuhi (misalnya nilai >= 70), sisa dari rantai *else if* akan diabaikan dan program langsung keluar dari struktur. Mekanisme *short-circuiting* ini menjamin setiap input hanya diklas*if*ikasikan ke dalam satu kategori, yang sangat penting untuk akurasi dan performa, terutama pada sistem yang memiliki banyak tingkatan kondisi [1].

2.5 Kondisi Multipel Alternatif (*switch-case*)

2.5.1 Kode program

```
package pekan4;

import java.util.Scanner;

public class NamaBulan {

    public static void main(String[]args) {


        Scanner scanner= new Scanner(System.in);

        System.out.print("Masukkan angka bulan(1-12): ");

        int bulan = scanner.nextInt();
```

```
switch (bulan) {
        case 1:
                System.out.println("Januari");
                break;
        case 2:
                System.out.println("Februari");
                break;
        case 3:
                System.out.println("Maret");
                break;
        case 4:
                System.out.println("April");
                break;
        case 5:
                System.out.println("Mei");
                break;
        case 6:
                System.out.println("Juni");
                break;
        case 7:
                System.out.println("Juli");
                break;
        case 8:
```

```
                System.out.println("Agustus");
                break;
            case 9:
                System.out.println("September");
                break;
            case 10:
                System.out.println("Oktober");
                break;
            case 11:
                System.out.println("November");
                break;
            case 12:
                System.out.println("Desember");
                break;
            default:
                System.out.println("Angka tidak valid");
            }
            scanner.close();
        }
}
```

2.5.2 Langkah kerja

1) Buat berkas NamaBulan.java dan terima input angka bulan.
2) Gunakan pernyataan *switch* dengan variabel bulan sebagai pemilih.

3) Tuliskan blok *case* untuk setiap nilai konstan (1 hingga 12).
4) Di akhir setiap blok *case*, sertakan pernyataan break untuk menghentikan eksekusi.
5) Tambahkan pernyataan default untuk menangani input angka yang tidak valid.

2.5.3 Analisis hasil

Pernyataan *switch* adalah alternatif yang jauh lebih rapi dan jelas untuk menguji suatu variabel terhadap banyak nilai konstan diskret [1]. Struktur ini cocok sekali untuk pemetaan nilai satu-ke-satu seperti konversi angka bulan ke nama bulan. Bagian yang paling krusial dalam struktur ini adalah pernyataan break. Jika break dihilangkan, kontrol akan "jatuh melalui" (*fall-through*), dan program akan mengeksekusi instruksi pada *case* berikutnya, yang dapat menyebabkan hasil yang tidak diinginkan [1]. Sementara itu, blok default berfungsi layaknya blok *else*, menangkap dan menangani semua input yang gagal dicocokkan oleh *case* mana pun.

BAB III

PENUTUP

3.1 Kesimpulan

Praktikum Pekan 4 mengenai Pernyataan Kondisi (*Conditional Statement*) telah memberikan pemahaman mendalam tentang cara kerja alur kontrol dalam program. Dari seluruh implementasi kode dan analisis yang telah dilakukan, kami dapat menarik beberapa poin kunci terkait pentingnya struktur kondisi:

1. Alur Program: Penggunaan struktur kondisi seperti *if* dan *if-else* adalah langkah fundamental untuk mengubah program dari sekadar menjalankan perintah sekuensial menjadi sistem yang dinamis. Penggunaan *if-else* secara khusus memastikan bahwa kode tidak akan "diam" saat syarat tidak terpenuhi, karena selalu ada respons pasti (jalur *true* atau *false*).
2. Efisiensi: Untuk kasus logika berjenjang, seperti sistem konversi nilai, penggunaan struktur *if-else if-else* adalah metode yang paling efisien. Hal ini disebabkan oleh fitur short-circuiting, yang menjamin pengujian akan langsung berhenti setelah menemukan kondisi yang benar. Mekanisme ini mencegah pengujian yang tidak perlu, sehingga menghemat waktu eksekusi [1].
3. Rapi: Pernyataan *switch-case* menawarkan solusi yang lebih rapi dan mudah dibaca (terutama untuk perbandingan nilai tunggal) dibandingkan rantai *if-else if* yang panjang. Kunci agar *switch* bekerja sesuai harapan adalah penggunaan kata kunci break, yang berfungsi vital untuk menghentikan eksekusi dan mencegah *fall-through* ke *case* berikutnya [1].

Secara keseluruhan, menguasai berbagai struktur kondisi ini merupakan lompatan krusial. Kompetensi ini membekali kami untuk merancang algoritma yang adaptif dan mampu merespons secara cerdas terhadap setiap variasi data input.

DAFTAR PUSTAKA

[1] P. Deitel dan H. Deitel, Java How to Program, Early Objects, 11th ed. Boston, MA: Pearson, 2018